**Modul 2 PBO Teori\_Ridhatul Nafsi**

**Pengenalan Bahasa JAVA**

**Latar Belakang**

**1.1 Sejarah Singkat JAVA**

Pada 1991, sekelompok insinyur Sun dipimpin oleh Patrick Naughton dan James Gosling ingin merancang bahasa komputer untuk perangkat konsumer seperti cable TV Box. Karena perangkat tersebut tidak memiliki banyak memori, bahasa harus berukuran kecil dan mengandung kode yang liat. Juga karena manufaktur – manufaktur berbeda memilih processor yang berbeda pula, maka bahasa harus bebas dari manufaktur manapun. Proyek diberi nama kode ”Green”. Kebutuhan untuk fleksibilitas, kecil, liat dan kode yang netral terhadap platform mengantar tim mempelajari implementasi Pascal yang pernah dicoba. Niklaus Wirth, pencipta bahasa Pascal telah merancang bahasa portabel yang menghasilkan intermediate code untuk mesin hipotesis. Mesin ini sering disebut dengan mesin maya (virtual machine). Kode ini kemudian dapat digunakan di sembarang mesin yang memiliki interpreter. Proyek Green menggunakan mesin maya untuk mengatasi isu utama tentang netral terhadap arsitektur mesin. Produk pertama proyek Green adalah Star 7 (\*7), sebuah kendali jarak jauh yang sangat cerdas. Dikarenakan pasar masih belum tertarik dengan produk konsumer cerdas maka proyek Green harus menemukan pasar lain dari teknologi yang diciptakan. Pada saat yang sama, implementasi WWW dan Internet sedang mengalami perkembangan pesat. Di lain pihak, anggota dari proyek Green juga menyadari bahwa Java dapat digunakan pada pemrograman internet, sehingga penerapan selanjutnya mengarah menjadi teknologi yang berperan di web. Java telah mengakomodasi hampir seluruh fitur penting bahasa – bahasa pemrograman yang ada semenjak perkembangan komputasi modern manusia :

1. Dari SIMULA, bahasa pada tahun 65-an, bahasa yang paling mempengaruhi Java sekaligus C++. Dari bahasa ini diadopsi bentukan – bentukan dasar dari pemrograman berorientasi objek.
2. Dari LISP – bahasa tahun 55-an. Diadopsi fasilitas garbage collection, serta kemampuan untuk meniru generic list processing, meski fasilitas ini jarang yang memanfaatkannya.
3. Dari Algol – bahasa pada tahun 60-an, diambil struktur kendali yang dimilikinya.
4. Dari C++, diadopsi sintaks, sebagian semantiks dan exception handling
5. Dari bahasa Ada, diambil strongly type, dan exception handling.
6. Dari Objective C, diambil fasilitas interface.
7. Dari bahasa SmallTalk, diambil pendekatan single-root class hiérarchie, dimana objek adalah satu kesatuan hirarki pewarisan
8. Dari bahasa Eiffel, fasilitas assertion yang mulai diterapkan di sebagian JDK 1.4

**1.2 Apa itu Teknologi Java ?.**

Sebagai sebuah bahasa pemrograman, Java dapat membuat seluruh bentuk aplikasi, desktop, web dan lainnya, sebagaimana dibuat dengan menggunakan bahasa pemrograman konvensional yang lain. Java adalah bahasa pemrograman yang berorientasi objek (OOP) dan dapat dijalankan pada berbagai platform sistem operasi. Perkembangan Java tidak hanya terfokus oada satu sistem operasi, tetapi dikembangkan untuk berbagai sistem operasi dan bersifat open source

**1.3 Mengapa Mempelajari JAVA?**

Berdasarkan white paper resmi dari SUN, Java memiliki karakteristik berikut :

1. Sederhana (Simple) Bahasa pemrograman Java menggunakan Sintaks mirip dengan C++ namun sintaks pada Java telah banyak diperbaiki terutama menghilangkan penggunaan pointer yang rumit dan multiple inheritance. Java juga menggunakan automatic memory allocation dan memory garbage collection.
2. Berorientasi objek (Object Oriented) Java mengunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Pemrograman berorientasi objek memodelkan dunia nyata kedalam objek dan melakukan interaksi antar objek-objek tersebut.
3. Terdistribusi (Distributed) Java dibuat untuk membuat aplikasi terdistribusi secara mudah dengan adanya libraries networking yang terintegrasi pada Java.
4. Interpreted Program Java dijalankan menggunakan interpreter yaitu Java Virtual Machine (JVM). Hal ini menyebabkan source code Java yang telah dikompilasi menjadi Java bytecodes dapat dijalankan pada platform yang berbeda-beda.
5. Robust Java mempuyai reliabilitas yang tinggi. Compiler pada Java mempunyai kemampuan mendeteksi error secara lebih teliti dibandingkan bahasa pemrograman lain. Java mempunyai runtime-Exception handling untuk membantu mengatasi error pada pemrograman.
6. Secure Sebagai bahasa pemrograman untuk aplikasi internet dan terdistribusi, Java memiliki beberapa mekanisme keamanan untuk menjaga aplikasi tidak digunakan untuk merusak sistem komputer yang menjalankan aplikasi tersebut.
7. Architecture Neutral Program Java merupakan platform independent. Program cukup mempunyai satu buah versi yang dapat dijalankan pada platform berbeda dengan Java Virtual Machine.
8. Portable Source code maupun program Java dapat dengan mudah dibawa ke platform yang berbeda-beda tanpa harus dikompilasi ulang.
9. Performance Performance pada Java sering dikatakan kurang tinggi. Namun performance Java dapat ditingkatkan menggunakan kompilasi Java lain seperti buatan Inprise, Microsoft ataupun Symantec yang menggunakan Just In Time Compilers (JIT).
10. Multithreaded Java mempunyai kemampuan untuk membuat suatu program yang dapat melakukan beberapa pekerjaan secara sekaligus dan simultan.
11. Dynamic Java didesain untuk dapat dijalankan pada lingkungan yang dinamis. Perubahan pada suatu class dengan menambahkan properties ataupun method dapat dilakukan tanpa menggangu program yang menggunakan class tersebut.

**1.4.Sebagian Fitur dari JAVA**

**1.4.1 Java Virtual Machine (JVM)**

JVM adalah sebuah mesin imajiner (maya) yang bekerja dengan menyerupai aplikasi pada sebuah mesin nyata. JVM menyediakan spesifikasi hardware dan platform dimana kompilasi kode Java terjadi. Spesifikasi inilah yang membuat aplikasi berbasis Java menjadi bebas dari platform manapun karena proses kompilasi diselesaikan oleh JVM.

**1.4.2 Garbage Collection**

Program Java melakukan garbage collection yang berarti program tidak perlu menghapus sendiri objek – objek yang tidak digunakan lagi. Fasilitas ini mengurangi beban pengelolaan memori oleh pemrogram dan mengurangi atau mengeliminasi sumber kesalahan terbesar yang terdapat pada bahasa yang memungkinkan alokasi dinamis.

**1.4.3 Code Security**

Code Security terimplementasi pada Java melalui penggunaan Java Runtime Environment (JRE). Java menggunakan model pengamanan 3 lapis untuk melindungi sistem dari untrusted Java Code.

1. Pertama, class-loader menangani pemuatan kelas Java ke runtime interpreter. Proses ini menyediakan pengamanan dengan memisahkan kelas – kelas yang berasal dari local disk dengan kelas – kelas yang diambil dari jaringan. Hal ini membatasi aplikasi Trojan karena kelas – kelas yang berasal dari local disk yang dimuat terlebih dahulu.
2. Kedua, bytecode verifier membaca bytecode sebelum dijalankan dan menjamin bytecode memenuhi aturan – aturan dasar bahasa Java.
3. Ketiga, manajemen keamanan menangani keamanan tingkat aplikasi dengan mengendalikan apakah program berhak mengakses sumber daya seperti sistem file, port jaringan, proses eksternal dan sistem windowing.

*Thank you !*

*Pembuat*

Ridhatul Nafsi